

# A Fast and Accurate Analytical Model of TCP Performance

D.F. Kassa and A.E. Krzesinski  
Department of Computer Science  
University of Stellenbosch  
7600 Stellenbosch  
South Africa

Tel: +27 21 808 4243 Fax: +27 21 808 4416

Email: debessay@cs.sun.ac.za

**Abstract**—The majority of traffic on the Internet uses the Transmission Control Protocol (TCP) as a transport layer protocol for the end-to-end control of information transfer. Accurate models of TCP performance are a key and basic step for designing, dimensioning and planning IP (Internet Protocol) networks. Packet-level simulation models do not scale with the growth of network capacities and number of users. Measurements also become very costly and inflexible with the growth and complexity of the Internet. This study proposes a simple, fast and accurate analytical model of TCP. The model gives Internet performance metrics, assuming that only basic network parameters such as the network topology, the number of users, link capacity, distance between network nodes and router buffer sizes are known.

The TCP performance model derives performance metrics which express the network quality of service. To obtain the performance metrics, TCP and network sub-models are used. A closed network of  $G/\infty$  queues is used to develop each TCP sub-model where each queue represents a state of a TCP connection. An  $M/M/1/K$  queue is used for each network sub-model which represents the output interface of an IP router with a buffer capacity of  $K$  packets. The two sub-models are iteratively solved.

Based on comparisons of our results with *ns2* simulation experiments and with other results from a known TCP model, our model is accurate, fast and simple.

**Index Terms**—Analytical models, quality of service, TCP, TCP Tahoe.

## I. INTRODUCTION

TCP-Tahoe is one of the most commonly used variants of TCP and is shown to out-perform the other variants of TCP (see [8]). The congestion caused by bulk transfer of files and FTP down-loads (called greedy or persistent TCP connections) is considerable. Therefore modeling greedy TCP-Tahoe connections gives a good estimation of Internet performance metrics.

This paper presents an analytical model to estimate the performance of greedy TCP-Tahoe connections. Where no confusion can arise, throughout this paper we use the term TCP to refer to greedy TCP Tahoe connections.

This work is supported by grant numbers 2054027 and 2677 from the South African National Research Foundation, Siemens Telecommunications and Telkom SA Limited.

The model is composed of an aggregate of several TCP and network sub-models. Each TCP sub-model represents a homogeneous group of TCP connections which refer to TCP connections sharing common characteristics (the same TCP version, comparable round trip times, similar loss probability, the same value of maximum window size expressed in packets and the like). The network sub-models represent the multi-bottleneck links traversed by the TCP connections with one network sub-model for each bottleneck link.

Each TCP sub-model receives an estimate of the packet loss probability along the TCP connection routes, as well as estimates of the queueing delays (RTT) at routers as inputs from the network sub-models. It then produces estimates of the load generated by the TCP connections in the group (TCP sub-model) for the network sub-model(s).

The rest of the paper is organized as follows. The TCP and network sub-models are presented in sections II and III respectively. The model complexity is described in section IV. The network topology used to validate our model is given in section V and comparison with *ns2* and another analytical model is given in and VI. Finally summary of the paper is given in section VII.

## II. THE TCP SUB-MODEL

As in [2], [3], [5] we model the dynamics of TCP as described in [6] using queueing theory.

The TCP model presented in this paper uses the notion [1] of modeling TCP as a Markov chain together with some formulas presented in [3] and [5]. This methodology yields a simple and accurate model of TCP dynamics.

TCP is modeled as a closed network of  $G/\infty$  queues, where each queue represents a state of the TCP protocol and each customer represents an active TCP connection.

The number of customers at a queue is the number of TCP connections that are in the corresponding state. As there is no limit to the number of connections in a given TCP state, each queue is modeled as an infinite server. The service times are given by a general distribution which depends on the average round trip time (RTT) experienced by TCP flows and TCP timeouts. The arrival process to the queues need

not be specified as only average arrival rates to the states are needed to solve the queueing network and derive the metrics of interest.

Where no confusion can arise, through out the paper we use the terms state and queue interchangeably as each queue in the model represents a state of a TCP connection.

The load generated by the TCP sub-model is derived as follows. We first identify six TCP states. The transition probabilities among the states are calculated using the *SS* and *CA* window size distributions and other closed form formulas. Closed form expressions are used to find the probabilities that TCP is in a given state with a specific window size value. These probability values are the relative number of visits to each queue with specific window sizes. Using these relative visit counts and a retransmission probability distribution, the effects of exponential back-off are obtained. The service times, the times each TCP-connection spends in each state are calculated. Using the service times and the relative number of visits, the mean value analysis (MVA) algorithm is used to obtain the average number of connections from which the normalized arrival rate to each queue with a specific window size is estimated. These arrival rates along with the average number of packets offered by each queue with a specific window size are then used to calculate the load offered to the network sub-model. The *ssthresh* is also calculated from the arrival rates.

#### A. The states of TCP connections

Analytical models of greedy flows usually disregard the initial transient phase, assuming that the network quickly operates at a steady-state (see chapter 20.7 of [6]). The states which a TCP connection assumes in a steady-state are

- *SS* which models the TCP slow start state. This refers to both transmission and retransmission of packets in *SS*.
- *SSF* which models the state where TCP waits for triple duplicate ACKs when losses occur during *SS*.
- *SST* which models the state where TCP waits for a timeout to expire when losses occur during *SS*. This refers to both transmission and retransmission timeouts.
- *CA* which models the congestion avoidance state.
- *CAF* which models losses during *CA* that trigger a fast retransmit.
- *CAT* which models the detection of losses by timeout during *CA*. This refers to transmission timeouts only as retransmission timeouts where the window size is less than or equal to 3 are modeled by the *SST* state.

Figure 1 presents a state transition diagram which is also the queueing network model of TCP-Tahoe as each state is a queue. Each node represents a state of a TCP connection. TCP starts transmission in the *SS* state and remains in that state increasing the *cwnd* value by 1 segment for each successful transmission of a segment thus doubling the *cwnd* value every *RTT* until *ssthresh* is reached. If an ACK signalling success of the segment is not received within one *RTT* then TCP either goes to the *SST* state where it waits for a timeout (TO) to expire or to the *SSF* state where it waits for a triple

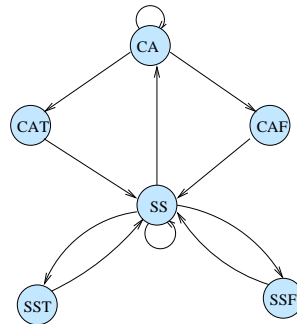


Fig. 1. The state transition diagram of greedy TCP-Tahoe

duplicate ACKs (TD). When *ssthresh* is reached, TCP moves into the *CA* state and it remains in that state increasing the *cwnd* by  $1/cwnd$  for every ACK received thus increasing the *cwnd* value by 1 every RTT. If TCP is in the *SSF*, *SST*, *CAT* or *CAF* states, it goes back into the *SS* state where the window size is reduced to 1 (TCP-Tahoe) to re-transmit the lost packet. The other transitions can be similarly explained. To derive the transition probabilities between the 6 states, we will use the symbols and formulas explained in the following section.

#### B. The packet loss probabilities

Let  $P_L$  denote the packet loss probability obtained from the network sub-model, let  $P_{L_f}$  denote the loss probability of the first packet in the active sliding window and let  $P_{L_a}$  denote the loss probability of any other packet of the same window.

As explained in [2]–[4] TCP introduces correlations in packet losses that have an impact on performance and consequently on any TCP modeling process. The loss probabilities  $P_{L_f}$  and  $P_{L_a}$  describe the correlation among losses of packets within the same congestion window where  $P_{L_a} = \alpha P_{L_f}$  and  $1 \leq \alpha \leq 1/P_{L_f}$ . As verified in [2]  $\alpha$  is set to  $\bar{w}$  (the average congestion window size computed from the TCP model) to model the increased correlation with the growth of the window size. Due to lack of space the derivation of  $P_{L_f}$  is omitted – see Appendix A of [7].

#### C. The transition probabilities

Let  $\mathcal{T} = \{SS, SST, SSF, CA, CAF, CAT\}$  denote the set of TCP states. Consider a state  $A_i \in \mathcal{T}$  and  $a_{min} \leq i \leq a_{max}$  where  $i$  denotes the size of the congestion window. The values of  $a_{min}$  and  $a_{max}$  are given in Table I where  $W_m$  is the maximum window size in packets.  $W_m = 64$  packets in this study. Let  $P(A_i, B_j)$  denote the probability

	<i>SS</i>	<i>SST</i>	<i>SSF</i>	<i>CA</i>	<i>CAF</i>	<i>CAT</i>
$a_{min}$	1	1	4	2	4	2
$a_{max}$	$W_m/2$	$W_m/2$	$W_m/2$	$W_m$	$W_m$	$W_m$

TABLE I

THE MAXIMUM AND MINIMUM WINDOW SIZE VALUES IN EACH STATE

that TCP goes from state  $A$  where  $cwnd = i$  to state  $B$  where  $cwnd = j$ . Clearly  $P(SST_i, SS_j), P(SSF_i, SS_j), P(CAF_i, SS_j)$

and  $P(CAT_i, SS_j)$  are all equal to 1 as TCP-Tahoe goes back to the  $SS$  state every time a loss occurs. Due to lack of space the derivation of the remaining  $P(A_i, B_j)$  is omitted – see Appendix B.3. of [7].

#### D. The $SS$ and $CA$ window size distributions

Let  $\pi(A)$  and  $\pi(A_i)$  denote respectively the probability that TCP is in state  $A$  and in state  $A_i$ .

In  $SS$  even though TCP doubles every  $RTT$  it grows in a continuous way as shown in [2]. Hence before growing to 4, the window must assume the value 3 and keep such value for at least 2 transmission times.

As defined above, the transition probability  $P(SS_i, SS_{i+1})$  is the probability of a successful increment of the window size  $i$  by 1 in  $SS$ . The probability  $P_{SS_i}$  of failure to increase the window size  $i$  by 1 in  $SS$  due to packet loss or because  $ssthresh$  is reached is

$$P_{SS_i} = \begin{cases} 1 & i = W_m/2 \\ 1 - P(SS_i, SS_{i+1}) & 1 \leq i < W_m/2. \end{cases} \quad (1)$$

Similarly the transition probability  $P(CA_i, CA_{i+1})$  is the probability of successful increment of the window size  $i$  by 1 in  $CA$ . The probability  $P_{CA_i}$  of failure to increase the window size  $i$  by 1 in  $CA$  is given by

$$P_{CA_i} = \begin{cases} 1 & i = W_m \\ 1 - P(CA_i, CA_{i+1}) & 2 \leq i < W_m. \end{cases} \quad (2)$$

The  $SS$  window size distribution is given by

$$\begin{aligned} P(i | SS) &= P(cwnd = i | \text{TCP is in state } SS) \\ &= \prod_{w=1}^{i-1} P(SS_w, SS_{w+1}) P_{SS_i}. \end{aligned} \quad (3)$$

for  $2 \leq i \leq W_m/2$  and  $P(1 | SS) = P_{SS_1}$  where  $P_{SS_i}$  is given in Equation 1 above.

The  $CA$  window size distribution is based on the fact that for long lived TCP connections at equilibrium, the  $ssthresh$  is reached quickly that TCP spends most of the time in the  $CA$  states. This assumption does not consider the few transitions from  $SS$  into  $CA$  in the derivation of the  $CA$  window size distribution. However these transitions from  $SS$  into  $CA$  are accounted for in the derivation of the stationary probability that TCP is in a given state as shown in the next section. Numerical results given in chapter 7 of [7] show that the above assumption has a small effect only in the  $cwnd$  size and  $ssthresh$  distributions for a small number of active TCP connections sharing the same link. Hence the  $CA$  window size distribution is given by

$$\begin{aligned} P(i | CA) &= P(cwnd = i | \text{TCP is in state } CA) \\ &= \prod_{w=2}^{i-1} P(CA_w, CA_{w+1}) P_{CA_i} \end{aligned} \quad (4)$$

for  $3 \leq i \leq W_m$  and  $P(2 | CA) = P_{CA_2}$  where  $P_{CA_i}$  is given in Equation 2 above.

#### E. The equilibrium probabilities

The transition probabilities  $P(A, B)$  are computed as

$$\begin{aligned} P(A, B) &= \frac{\sum_{i=a_{min}}^{a_{max}} \sum_{j=a_{min}}^{a_{max}} \frac{\pi(A_i) P(A_i, B_j)}{\pi(A)}}{\sum_{i=a_{min}}^{a_{max}} \sum_{j=a_{min}}^{a_{max}} P(i | A) P(A_i, B_j)} \quad (5) \end{aligned}$$

where the  $P(A_i, B_j)$  are given in Appendix B.3 of [7].

The stationary probabilities  $\pi(B)$  that a TCP connection is in state  $B \in \mathcal{T}$ , which is the set of TCP states described in section II-A, are computed as

$$\begin{aligned} \pi(CA) &= \pi(SS)P(SS, CA) + \pi(CA)P(CA, CA) \\ &= \frac{\pi(SS)P(SS, CA)}{1 - P(CA, CA)}. \end{aligned}$$

Since TCP has to be in one of the 6 states at a given time

$$\pi(SS) + \pi(SST) + \pi(SSF) + \pi(CA) + \pi(CAT) + \pi(CAF) = 1. \quad (6)$$

Thus

$$\begin{aligned} &\pi(SS)(1 + P(SS, SST) + P(SS, SSF)) \\ &+ \frac{\pi(SS)P(SS, CA)}{1 - P(CA, CA)} (1 + P(CA, CAT) + P(CA, CAF)) = 1 \\ &\pi(SS) \left( 1 + P(SS, SST) + P(SS, SSF) \right. \\ &+ \left. \frac{P(SS, CA)}{1 - P(CA, CA)} (1 + (1 - P(CA, CA))) \right) = 1 \\ &\pi(SS) \left( 1 + P(SS, SST) + P(SS, SSF) + P(SS, CA) \right. \\ &+ \left. \frac{P(SS, CA)}{1 - P(CA, CA)} \right) = 1 \\ &\pi(SS) \left( 1 + (1 - P(SS, SS)) + \frac{P(SS, CA)}{1 - P(CA, CA)} \right) = 1. \end{aligned}$$

Therefore

$$\pi(SS) = \frac{1 - P(CA, CA)}{P(SS, CA) + (2 - P(SS, SS))(1 - P(CA, CA))}.$$

The  $\pi(B_i)$  can now be obtained as follows.

$$\begin{aligned} \pi(SS_i) &= \pi(SS) P(i | SS), \quad 1 \leq i \leq W_m/2 \\ \pi(CA_i) &= \pi(CA) P(i | CA), \quad 2 \leq i \leq W_m \end{aligned}$$

TCP offers 2 packets in each of the  $SS_i$ ,  $2 \leq i \leq W_m/2$  states and one packet in  $SS_1$ . The loss of the first packet sent in  $SS_i$  is detected either by timeout (TO) or by triple duplicate ACKs (TD) when the window size has grown to  $2i - 2$  provided  $ssthresh$  is not reached on. The loss of the second packet sent in  $SS$  when the window size is  $i$  is detected when the window size is  $2i - 1$  provided  $ssthresh$  is not reached on.

For example consider  $ssthresh \geq 4$ . If packet 1 is sent and ACK 1 returns after one  $RTT$ ,  $cwnd = 2$  and two packets 2 and 3 are sent back-to-back. If ACK 2 doesn't return, the loss of packet 2 is detected when  $cwnd = 2 \times 2 - 2 = 2$ . If ACK 2 returns after one  $RTT$ ,  $cwnd = 3$  and packets 4 and 5 are sent. If at this stage ACK 3 doesn't return, it means that the loss of packet 3 is detected when  $cwnd = 2 \times 2 - 1 = 3$ . Thus

the loss which is detected when the window size is  $i$  provided  $ssthresh$  is not reached until the window grows to  $i$  is the loss of the first packet which was sent when the window size was  $(i+2)/2$  if  $i$  is even or the loss of the second packet which was sent when the window size was  $(i+1)/2$  if  $i$  is odd.

On the other hand even though the losses of the first and second packets sent when the window size was  $(i+2)/2 + k$ ,  $0 \leq k \leq (i-2)/2$ ,  $i$  being even and  $(i+1)/2 + k$ ,  $0 \leq k \leq (i-1)/2$ ,  $i$  being odd respectively are detected when the window size is  $2((i+1)/2 + k) - 1 = 2((i+2)/2 + k) - 2 = i + 2k$ , transitions from  $SS_{(i+2)/2+k}$  and  $SS_{(i+1)/2+k}$  into  $SST_i$  or  $SSF_i$  occur if there is a loss and  $ssthresh$  is reached when  $cwnd = i$ .

Therefore for  $1 \leq i \leq W_m/2$

$$\pi(SST_i) = \sum_{j=a}^i \pi(SS_j) P(SS_j, SST_i)$$

and for  $4 \leq i \leq W_m/2$

$$\pi(SSF_i) = \sum_{j=a}^i \pi(SS_j) P(SS_j, SSF_i)$$

where

$$a = \begin{cases} (i+1)/2, & i \text{ is odd} \\ (i+2)/2, & i \text{ is even.} \end{cases}$$

A transition into  $CAT_2$  occurs only from  $CA_2$ . Hence

$$\pi(CAT_2) = \pi(CA_2) P(CA_2, CAT_2).$$

For all other values of  $i$  a transition into  $CAT_i$  or  $CAF_i$  occurs if either the first packet sent in  $CA_i$  is lost or the first packet sent in  $CA_{i-1}$  is successful and any of the other packets sent in  $CA_{i-1}$  is the first lost packet. For details on these transitions see Appendix B.3.3 of [7].

Therefore for  $3 \leq i \leq W_m$

$$\pi(CAT_i) = \pi(CA_{i-1}) P(CA_{i-1}, CAT_i) + \pi(CA_i) P(CA_i, CAT_i)$$

and for  $4 \leq i \leq W_m$

$$\pi(CAF_i) = \pi(CA_{i-1}) P(CA_{i-1}, CAF_i) + \pi(CA_i) P(CA_i, CAF_i).$$

#### F. The effects of exponential back-off

Unlike the model of Garetto *et al.* [5], in our model the exponential back-off and the other retransmission states are built into the  $SS$  and  $SST$  states as shown in Figure 1 to reduce the complexity and to obtain simple closed form expressions for the  $SS$  and  $CA$   $cwnd$  size distributions.

After a timeout loss in both  $SS$  and  $CA$  TCP returns to  $SS$  where the timeout value is obtained using Karn's algorithm (see [6]) during each retransmission. Define the first transmission to be retransmission 0. A TCP connection which starts in  $SS$  with  $cwnd = 1$  goes to the  $(j+1)$ st retransmission where  $0 \leq j < C$  if the  $j$ th retransmission is not successful and stays in the  $j$ th retransmission if the  $j$ th retransmission is successful, where  $C$  is the maximum number of consecutive retransmissions allowed before closing the connection.  $C =$

16 for TCP-Tahoe. The probability that TCP is in the  $j$ th retransmission,  $0 \leq j \leq C$  is modeled as a geometric distribution with parameter  $P_{S_f} = 1 - P_{L_f}$  (probability that the first packet is successfully transmitted). Let  $P_{rt_j}$  denote the probability that TCP is in the  $j$ th retransmission. Then

$$P_{rt_j} = (1 - P_{S_f})^j P_{S_f}.$$

Let  $\pi(SST_i^j)$  be the probability that TCP is in the  $SST$  state when the window size is  $i$ ,  $1 \leq i \leq 3$  in retransmission  $j$ ,  $0 \leq j \leq C$ . Then

$$\pi(SST_i^j) = P_{rt_j} \pi(SST_i) \quad (7)$$

The effect of Karn's algorithm and exponential back-off ends and the retransmission index is reset to zero if the two newly transmitted packets in  $SS_2$  after the successful retransmission are also successful. If transmission of the first or only the second packet is not successful, TCP goes to the  $SST_2$  or  $SST_3$  state of that retransmission where the timeout value is obtained from the exponential back-off using Karn's algorithm. After this timeout value expires TCP goes back to  $SS_1$  of the same retransmission. Thus only the  $cwnd$  values 1, 2 and 3 are required to deal with TCP's exponential back-off. This is shown in figure 2 of [5] where the transmission and retransmission states are separately drawn.

The service times of each queue are described in the next section.

#### G. The service times

Let  $\tau(A_i)$  denote the time which TCP spends in state  $A_i$ . These service times and their derivations are given in Appendix C of [7] and Table 1 of [5] with the exception of  $\tau(SST_i)$  which is

$$\tau(SST_i) = \begin{cases} \sum_{j=0}^C \pi(SST_i^j) \tau(SST_i^j) & 1 \leq i \leq 3 \\ \tau(SST_i^0) & i > 3 \end{cases}$$

where  $\tau(SST_i^j)$  is service time of the  $SST$  state when  $cwnd = i$  in retransmission  $j$ . These values are also given in Appendix C of [7].

The initial timeout value  $T_0$  used in the calculation of  $\tau(SST_i^j)$  is given by  $T_0 = \max(3tick, 4\overline{RTT})$  as is the case in [3], [5]. The  $tick$  is the minimum amount of time that separates non-self clocked TCP protocol operations. For  $1 \leq i \leq 3$ ,  $\tau(SST_i)$  is the weighted average of the service times of the  $SST_i$ 's at each retransmission where the weights are the probabilities of being in retransmission  $j$  where  $0 \leq j \leq C$  of the corresponding window size. The same idea is used in the derivation of the load offered by  $SST_i$  given in section II-H below.

Using the service times and the relative visit counts explained above, the mean value analysis algorithm (MVA) is used to obtain the average number of connections in each queue from which the normalized arrival rate to each queue with a specific  $cwnd$  size is estimated. The MVA algorithm has simple form expressions for closed queueing network of infinite servers which is the case in our study.

At this stage the  $ssthresh$  is calculated as shown in Appendix B.3.2 of [7] or section A of [5].

### H. The load offered to the network sub-model

The aggregate packet load offered to the IP network by the TCP connections in the different states is

$$\Lambda = \sum_{A \in \mathcal{T}} \Lambda(A) = \sum_{A \in \mathcal{T}} \sum_{i=a_{min}}^{a_{max}} \lambda(A_i) L(A_i) \quad (8)$$

where  $\lambda(A_i)$  are the arrival rates of connections to state  $A_i$  obtained from the MVA. The  $L(A_i)$  is the number of packets offered from queue  $A_i$  to the underlying IP network. The  $L(A_i)$  are used in [5] and their derivations are given in Appendix D of [7] with the exception of  $L(SST_i)$  which is given by

$$L(SST_i) = \begin{cases} L(SST_i^0) & i = 1, 2, 4, 5, \dots \\ \sum_{j=0}^C \pi(SST_i^j) L(SST_i^j) & i = 3 \end{cases}$$

where  $L(SST_i^j)$  is the load offered by the  $SST$  state when the  $cwnd = i$  in retransmission  $j$ . These values are also given in Appendix D of [7].

### III. THE NETWORK SUB-MODEL

The network sub-model which focuses on the IP network receives the traffic load  $\Lambda$  packets/second collectively offered by the TCP connections in different states (queues) from the TCP sub-model. The network sub-model then uses an  $M/M/1/K$  queue with a router buffer capacity of  $K$  packets and link capacity of  $C$  packets/second (the load  $\rho = \Lambda/C$ ) to compute the expected number of customers  $E_N$  and the loss probability  $P_L$ . Even though the  $M/M/1/K$  assumes independent Poisson arrivals, the arrival rate given to our network sub-model ( $M/M/1/K$ ) is generated from our TCP sub-model(s) which take into account the correlations in network traffic and which stem from the closed-loop congestion control algorithms implemented at the transport layer.

### IV. THE COMPLEXITY OF THE MODEL

To find end-to-end TCP performance metrics the TCP and network sub-models are solved for each bottleneck link. Let the number of bottleneck links be  $L$  and the number of TCP sub-models for each bottleneck link be  $M$ . The complexity of our model for the multi-bottleneck network is  $O(LM)$  as the simple closed form formulas have to be used for each TCP sub-model at each bottleneck link.

The CPU time of the *ns2* simulation of our model increases dramatically with the increase of the number of active TCP connections, while it is almost independent of the number of active TCP connections for our model. For thousands of TCP connections *ns2* simulation experiments can not be computed on a PC. For a single TCP–Network sub-model pair, the Polito model (we refer to the model given in [5] as the Polito model) is 2 to 4 orders of magnitude faster than the *ns2* simulation. In the Polito model, a system of  $N_q$  linear equations has to be solved at each iteration of the fixed point procedure.  $N_q$  is the number of queues (states of a TCP connection) and depends on  $W_m$  the value of the maximum window size and  $C$  the number of consecutive retransmissions allowed before a TCP

connection closes. The complexity of Polito model for a multi-bottleneck (realistic) network setup is therefore  $O(LMN_q^3)$  although the almost triangular nature of the system of linear equations may reduce the complexity to  $O(LMN_q^2)$ .

Therefore the gain in CPU speed of our model is significant.

### V. THE APPLICATION OF THE MODEL

The network topology is presented in [2], [3], [5]. It closely resembles to the path followed by Internet connections from the Politecnico di Torino (Italy) LAN to sites in Europe and the USA. For these connections two scenarios, (1) when only the 5000 km long 45 Mb/sec MI-NY (Milan–NewYork) link is a bottleneck and (2) when both this MI-NY and the 100 km long 10 Mb/sec POLI-TO (Politecnico–Turin) links are bottlenecks are considered. The total length of connections crossing only the MI-NY single bottleneck link varies from 5,400 to 11,600 km and that of the connections which cross both bottleneck links varies from 5,351 to 8,960 km.

The user and ACK packet sizes are assumed to be 1024 bytes and 40 bytes respectively. Buffer sizes of 512 and 128 packets and the standard TCP tick value of 500 ms are considered.

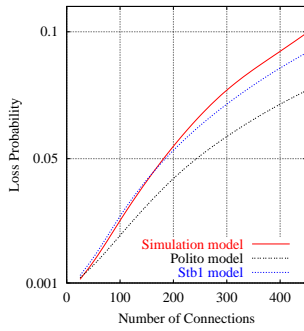
### VI. COMPARISON WITH THE POLITO MODEL AND NS2

Figure 2 compares the loss probability  $P_L$ , the average  $cwnd$  and  $ssthresh$  as computed by our model against the Polito model and *ns2* simulation experiment for the single bottleneck link. Figure 3 compares the loss probability obtained by our model against the Polito model and the *ns2* simulation experiments for the multi-bottleneck links.  $N1$  refers to the total number of active TCP connections sharing the POLI-TO link, 25% of which are assumed to cross the MI-NY bottleneck link.  $N2$  represents the total number of active TCP connections sharing the MI-NY link.  $N2 - 0.25N1$  of these connections are represented by a separate TCP sub-model and cross only the MI-NY single bottleneck link. If  $P_{L1}$  and  $P_{L2}$  are the packet loss probabilities of the MI-NY and POLI-TO links then the packet loss probability,  $P_L$  on the multi-bottleneck links is  $P_L = 1 - (1 - P_{L1})(1 - P_{L2}) = P_{L1} + P_{L2} - P_{L1}P_{L2}$ . Given the Polito network topology, this value is high as the packets have to cross both bottleneck links.

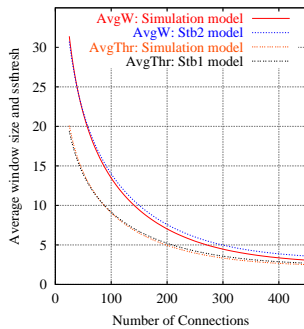
From the figures it can be seen that our model gives more accurate predictions of the loss probabilities than the Polito model. It also gives accurate estimations of the average  $cwnd$  and  $ssthresh$  values.

### VII. SUMMARY AND FUTURE WORK

This paper presents a closed queueing network model for the estimation of the performance of greedy TCP connections. The queueing network gives a detailed description of the behavior of TCP-Tahoe connections and their interaction with the underlying IP network. Numerical results for loss probabilities, average  $ssthresh$  and  $cwnd$  sizes as a function of the number of concurrent TCP connections for different buffer sizes were obtained. The results were compared with the *ns2* package. When compared with the model given in



(a) Packet loss probability of the MI-NY single bottleneck link



(b) Average window size and  $ssthresh$  of the MI-NY single bottleneck link

Fig. 2. Performance measures: the buffer capacity is 512 packets and the TCP  $tic$  value is 500 ms

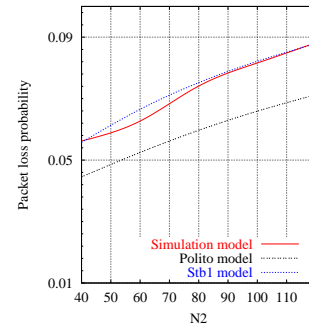
[5] (the Polito model) our model is faster and more accurate. The low computational complexity of our modeling approach will allow us to analyze thousands of connections and more complex networking scenarios which cannot be simulated with the *ns2* package.

We are extending the techniques used in this study to different TCP implementations, to short-lived TCP connections, to wireless links and to non-responsive UDP flows yielding a comprehensive, accurate and fast Internet model with many interacting TCP and UDP flows.

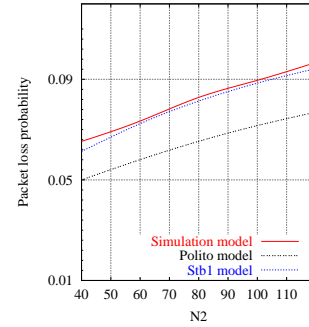
## REFERENCES

- [1] Å. Arvidsson, AE. Krzesinski. A Model of TCP Link. In *15th ITC Specialist Seminar on Internet Traffic Engineering and Traffic Management*, Wurzburg, Germany, July 2002.
- [2] M. Garetto, R. Lo Cigno, M.Meo, M.Ajmone Marsan. A Detailed and Accurate Closed Queueing Network Model of Many Interacting TCP Flows. In *IEEE INFOCOM 2001*, pages 1706-1715, Anchorage, Alaska, April 2001.
- [3] M.Garetto, R.Lo Cigno, M. Meo, M. Ajmone Marsan, Modeling short-lived TCP connections with open multi-class queueing networks, *COMPUTER NETWORKS*, pp.153-176, February 2004.
- [4] J. Padhye, V. Firoiu, D. Towsley, J. Kurose, Modeling TCP Reno performance: a simple model and its empirical validation, *IEEE/ACM Transactions on Networking* 8 (2) (2000) 133-145.

**Debessay F. Kassa** obtained the BSc in Mathematics from the University of Asmara, Eritrea and the BSc (Honours) in Computer Science from the University of Stellenbosch. He is currently an MSc student in Computer Science at the University of Stellenbosch.



(a)  $N1 = 32$



(b)  $N1 = 40$

Fig. 3. Packet loss probability of the MI-NY and POLI-TO multi-bottleneck links: the buffer capacity is 128 packets and the TCP  $tic$  value is 500 ms

- [5] M.Garetto, R.Lo Cigno, M.Meo, M. A. Marsan, Closed Queueing Networking Models of Interacting Long-Lived TCP Flows, *IEEE/ACM Transactions on Networking*, Vol.12, No.2, pp. 300-311, ISSN: pp. 1063-6692, April 2004.
- [6] W.R.Stevens. *TCP/IP Illustrated, Vol. 1*. Addison Wesley, Reading, MA, USA, 1994.
- [7] D. F. Kassa, Fast and Accurate Closed Queueing Network Models of TCP Performance, M.Sc. Thesis, Department of Computer Science, University of Stellenbosch.
- [8] B. Sikdar, S. Kalyanaraman and K. S. Vastola, "Analytic Models for the Latency and Steady-State Throughput of TCP Tahoe, Reno and SACK," *IEEE/ACM Transactions on Networking*, vol. 11, no. 6, pp. 959-971, December 2003.